AMBOSS: A Task Modeling Approach for Safety-Critical Systems

Matthias Giese[1], Tomasz Mistrzyk[2], Andreas Pfau[1], Gerd Szwillus[1], and Michael von Detten[1]

[1] University Paderborn, Institute for Computer Science, 33102 Paderborn, Germany
[2] OFFIS e. V., 26121 Oldenburg, Germany
{flipsi,andypf,szwillus,austin}@upb.de, Tomasz.Mistrzyk@offis.de

Abstract. In a recent project we created AMBOSS, a task modeling environment taking into account the special needs for safety-critical socio-technical systems. An AMBOSS task model allows the specification of relevant information concerning safety aspects. To achieve this we complemented task models with additional information elements and appropriate structures. These refer primarily to aspects of timing, spatial information, and communication. In this paper we give an introductory overview about AMBOSS and its contribution to modeling safety-critical systems. In addition, we present AmbossA, the visual pattern language for detecting particular constellations of interest within a task model.

Keywords: task modeling, safety-critical systems, socio-technical systems, task editor, simulation, task patterns.

1 Introduction

Task modeling approaches such as ([12], [9], [17], [2], and [5]) primarily concentrate on the hierarchical decomposition of tasks into subtasks and their relative ordering with temporal relations. Task models are typically used in the early phases of system design or as documentation method for the result of task analyses. The models are semi-formal in nature in the sense that they formally structure (hierarchy, temporal relations) informal elements (task descriptions). They have been used in system design ([10], [8]) or user interface design ([14], [6], [18]), but are also used increasingly for the design and analysis of workplace situations [17].

An important concept in most task modeling approaches concerns the actors performing the tasks. In CTTE [12], for example, explicitly identified roles co-operate in the concurrent performance of a highly structured task. An appropriate role or user model is included in all approaches mentioned above. Using these models, it is possible to describe task execution in complex socio-technical systems. These systems on one hand include technical components, such as computers, transport devices, telecommunication devices and production machines, and on the other hand they incorporate human actors, who are performing manual tasks and interactive tasks, i.e. are using machines to support their work. Typical examples are, for instance, execution of

P. Forbrig and F. Paternò (Eds.): HCSE/TAMODIA 2008, LNCS 5247, pp. 98–109, 2008.

routine maintenance procedures in a hydro power plant, or blood sample processing in a hospital. Both examples are performed by a group of co-operating human actors who are working both with technical systems and without them.

Specifying a task model for such a system documents the order of and the rationale behind the planning and structuring of the tasks to be performed. This is useful for analyzing an existing socio-technical system, or to start the design of a new not yet existing process. If in addition the socio-technical system is a safety-critical system the task model can be helpful in detecting potential problems created by, for example, inadequate task order, disproportionate distribution of workloads between actors, or lack of time in critical phases of task execution.

In a recent project we created AMBOSS [1], a task modeling environment taking into account the special needs for safety-critical socio-technical systems. An AMBOSS task model allows the specification of risk assessment factors, barriers protecting human beings and material value from harm [7], and the information flow between tasks. AMBOSS allows the specification of relevant information concerning these and additional issues. In this paper, however, we want to focus on some aspects which have not yet been integrated with task modeling to that extent before and which are especially relevant for socio-technical safety-critical systems: timing, topology, and communication.

A correct timing of co-operative actions is frequently vital for the correct and safe execution of safety-critical processes. We deal with this aspect within the simulation component of AMBOSS, which takes into account preconditions for tasks and their explicit link with barriers sheltering man and material. An example from health care would be to make sure that the x-ray device is only switched on after the operator has left the room. In addition, AMBOSS allows the specification of the spatial behavior of a process, i.e. to say where some task is performed, which objects are available at this position, and whether objects are transferred from one room to another during task execution. As an example consider the fact that it has a crucial influence on the end result whether a patient is in the neighboring room or far away from all nurses. While both time and space may be irrelevant for a large number of task models, as they are either trivial or self-evident, there are important cases of safety-critical processes depending on this type of information.

This applies even more so to the aspect of communication between the actors in a task model. Special emphasis has been given within AMBOSS to model possible variants of the properties of communication as needed for task execution. In co-operative processes it is often the case that an actor (man or machine) performing a subtask needs information from another actor performing another subtask. This implies task dependencies, with respect to task ordering, task triggering, and the quality of the communication between tasks. As an example, we analyzed a medication dosage process based on a blood sample, which included communication via hand-written notes, database queries, and casually uttered oral remarks between the actors.

Enriching a task-model with more detail burdens the designer or analyzer of a system, i.e. the AMBOSS user, with an increasingly complicated model which is hard to master. Task models tend to grow a lot when applied to real-world cases, and there is no use in specifying to great detail, when the user loses control over an outgrowing model. This is why we created a possibility to specify patterns of task model elements, and to automatically check their existence within a large task model. The

"AmbossA" language is a visual language which can be used to define "critical" or "interesting" patterns within task models and then apply a search function to detect or negate their existence in a model.

We will deal with the aspects mentioned in the subsequent chapters and conclude with a statement on the current state of the development. The features described here are all implemented in the current version, with the exception of the procedure for communication analysis, as sketched out in chapter 5. The current version of AMBOSS is freely available for non-commercial use for download on the AMBOSS website [1].

2 The Task Modeling Environment AMBOSS

AMBOSS is a task modeling environment following the traditional approach of hierarchical task structures with temporary relations between subtasks. Hence, AMBOSS provides the typical tree-based editing functions, such as adding a child node or a sister node; apart from that, however, we allow direct manipulation of nodes and connections for easy structure manipulation tasks. A task node, for instance, can be placed into the drawing area without being connected at all, and it can later on be linked to other nodes. Although liberal in the interaction style, structure constraints are ensured by checks in the background, inhibiting for instance the creation of cycles. The order of subtasks is defined by the relative placement around the parent node; hence the order can be modified by simply shifting the child nodes into the correct order. This concept has been implemented in K-MADe [3] as well, while CTT [12] and VTMB [5] are restricted to tree operations. A similar freedom of interaction is available in Tamot [9] without underlying checks being performed.

AMBOSS goes far beyond the traditional task modeling approach. Based on its implementation with plug-ins to the Eclipse framework, the editor provides flexible views on additional information necessary for modeling safety-critical systems. The elements included deal with roles, barriers, task objects, risk factors, and messages. As the Eclipse framework enables a flexible customized arrangement of views, the user is able to open, maximize, minimize, or close whatever view is needed for a specific analysis task.

AMBOSS allows the user to implement hierarchical role specifications in order to appoint the performing actor to a task and to stress its duties and abilities. Similar approaches were already mentioned in other environments ([12], [5], and [3]).

There are three basic types of actors in a socio-technical system: human, system, and abstract – the last describing tasks performed in co-operation between man and machine. In addition, we can specify subtypes of actors (such as physician or nurse) and instances of actors (such as the nurse Ms. Smith) as refinements of the roles human and system. When defining a role for a task this is consequently inherited by the subtasks and propagates up to the parent nodes in the task tree. In addition to being an actor within a task, a role can also be specified as being the person responsible for the task (such as for the correct treatment of a blood sample). A screenshot of the AMBOSS environment is shown in Fig. 1.

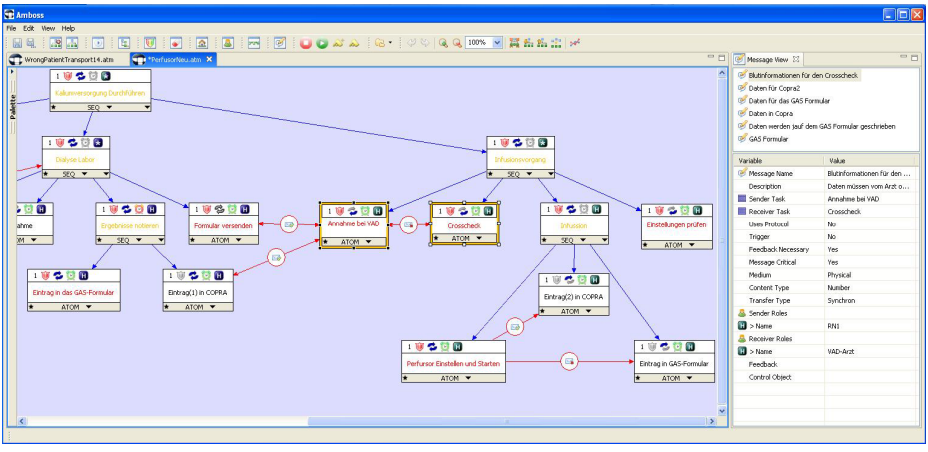

Fig. 1. The AMBOSS Environment

A concept showing up in several approaches for modeling safety-critical systems is the concept of barriers [7]. Barriers are means to prevent harm from human beings and material while operating a system; examples are the lead apron sheltering the patient and the operator from X-ray radiation, or the law against crossing a red traffic light. Barriers are linked in different ways to task performance: barriers can ensure that safety-critical tasks do not harm life or material; tasks can activate a barrier (such as putting on the lead apron), or deactivate it (e.g. by switching of a traffic light for maintenance). Hence, AMBOSS allows the specification of relations like these between barriers and tasks, and they are evaluated an can be "observed" during the simulation included in the environment (see chapter 3 below).

In addition, tasks can be rated as being critical. A numerical risk factor is assigned, based on estimations of probability, severity of consequences, and detectability on a scale from 1 to 10 [16]. By multiplying the numbers the overall factor is computed, which enables a summative evaluation of the risk involved. Another important influential factor when executing safety-critical tasks is timing. Hence, we allow the assignment of timing information to tasks, specifying minimal, maximal, or average task durations. This timing information is also used during the simulation of the models. In addition, AMBOSS incorporates a task object concept, which means that the user can specify which objects are modified by a task.

Once the model is defined, the user can start a simulation which takes into account the task hierarchy, temporal relations, conditions, barriers, message flow, and triggering through communication. The user can observe, which task (and actor) is sending what kind of information to whom, and whether or not triggering information has been sent or not. As the simulation also reflects the activation and deactivation of barriers, the user in every situation of the simulation can tell whether a necessary barrier is in place while a safety-critical task is executed. To compare and analyze different threads of execution one can save scenarios and watch their simulation repeatedly. Task model simulation has been done before ([9], [5], [3]) in part including the scenario feature mentioned as well, with the special case of K-MADe [3] which is

a particularly detailed and rich model. None of these, however, reflects the needs of safety-critical systems including the communication issues together with timing and spatial behavior as much as AMBOSS does.

3 Simulation of Timing and Conditions

The ability to simulate a task model is an essential tool in order to assure that the model behaves as intended and to control the behavior in a dynamic environment before the system is actually built. During the simulation the user chooses the tasks which he wants to start and to stop according to the choices presented by the simulator. These choices depend on the temporal relations between the tasks and are updated after each simulation step. Temporal relations are assigned to a task and control the course of action of its subtasks. AMBOSS contains six different temporal relations:

- SEQ: The subtasks are performed in a fixed sequence from left to right
- SER: The subtasks are performed in an arbitrary sequence
- PAR: The subtasks can start and stop in an inordinate way.
- SIM: All subtasks have to start before any subtask may stop.
- ALT: Exactly one subtask is performed.
- ATOM: This task has no subtasks.

In addition, the simulator of AMBOSS focuses on the integration of safety relevant aspects of the task model by controlling the state of the barriers and considering the flow of communication. Fig. 2 shows an AMBOSS simulator screen-shot. The system allows the user to view all parameters which influence the current task without switching back to AMBOSS task modeling view. As shown on the bottom left of Fig. 2 the user is informed about the objects being manipulated by the task, the rooms in which the task occurs (see chapter 4), the roles performing the task, the communication flow of the task and the barriers protecting the task from hazards. Furthermore the user is able to recognize at a glance which tasks are executed concurrently as shown in the middle area of Fig. 2. Task execution is symbolized by rectangles, where the ordering from left to right coincides with the order of performance. The user operates the simulator by starting and stopping tasks. This is done by double clicking the name of a task in the lists at the top left.

In order to guide the user's attention to the safety critical elements of a system it is possible to define certain preconditions for a task. If a task's precondition is not met the user will not be able to start it during the simulation. There are two different types of preconditions.

Message preconditions denote the concept that a message containing certain information is necessary to trigger a task, and allow its correct performance. The simulator keeps track of communication that has already taken place and enables the start of tasks accordingly. The user is kept informed about the progression of the simulation including the communication textually (see bottom right of Fig. 2). The simulator also checks for some mistakes being made during the modeling phase, such as if a message requires feedback but the feedback has not been defined yet.

The second type of preconditions refers to dynamical, “enactable” barriers. An enactable barrier is a barrier which does not achieve its protective impact all the time but has to be activated to do its service. This activation is being accomplished during the performance of a task. An example for this is X-ray protective clothing which has to be put on before it can render its service. During the simulation the user can check whether the simulated chain of action led to a situation in which the task can be performed safely.

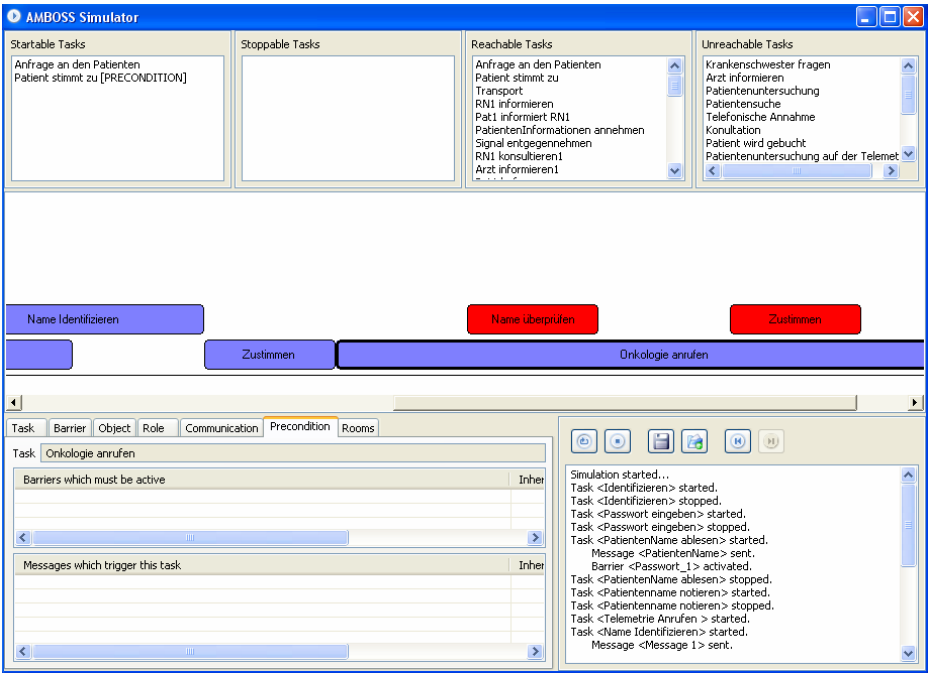

Fig. 2. AMBOSS Simulator Overview

Another feature of the AMBOSS simulator is the inclusion of scenarios. A scenario is an instance of the task model, i.e. one concrete task execution sequence to perform the complete task described. The user is able to save these scenarios and to compare the different effects of his decisions during the simulation.

With these possibilities it is possible to check safety relevant aspects of a task before a system is even built or to inspect the work flow of an existing system. The user can experiment with the different possible chains of events and compare the different effects of the chosen scenarios on the execution of the whole task. The simulator stresses the need to provide security for the performance of the tasks and incites the user to deal with potential hazards. This ultimately leads to an improvement of the way a task is performed and the elimination of the weak points of a system.

4 Spatial Behavior

In most of all considered models which represent safety critical systems or scenarios one of the important aspects is topology, i.e. a specification of the spatial relations between different task execution steps. This deals with the "positions" of both actors and material or objects, needed for the task. Frequently, there are situations where tasks are performed in a certain position of the work space (e.g. directly aside an x-ray device) which are harmless as long as the work space is "clean" (e.g. the x-ray device is off) and critical when the work space is "contaminated" (e.g. the x-ray device is working). To capture dependencies like these we introduced a rooms concept within AMBOSS. A room in this abstract sense can be a physical chamber in a building (e.g. an operating theatre), but it can also be a complete floor in a hospital, or an airplane hangar on an aircraft carrier. The limits of a room cannot be defined generally; they are subject to the abstraction process during modeling. For understanding of the concept, the idea of an actual room is a good start though.

A room in AMBOSS has different static properties, a unique name to identify the element, an informal textual description, a maximum number of persons that fit into a room, and a flag that indicates, whether this room is lockable or not.

More important are the dynamic properties, which specify the relationship between a room and the tasks, objects, roles and barriers.

To take into account that every task is performed at a certain place, the user can specify this relationship either from the point of view of the task or of the room. For every room the user can relate the tasks, which are performed within it. On the other hand, for every task it is possible to select the rooms, where the task takes place. One inherent property of the room concept is that every task is executed in at most one room.

Objects of the task model can be assigned to rooms in a similar bidirectional way. Objects, other than the task themselves, however, can be moved from one room to the other. So, when an object is specified, the user can specify whether the object is movable or not. If it is not, the user can relate it to one specific room, where this object is located (such as the x-ray device in the x-ray room). If the object is movable, the user has to specify, in which rooms the object can be located, and in which ones the object is not allowed because of procedural or factual restrictions (such as a free-to-move perfusor device, which is allowed in the patients' rooms but not in the operating room).

In some cases, it is important to express that not every person has access to a room. This is represented with the room-role-relationship. For every room the user can select the roles, which have either the permission to enter the room, a restricted kind of permission or no permission to enter.

During the modeling process, AMBOSS checks whether the user is about to create an invalid situation. For instance, when an object has already a fix assignment to one room, it cannot have an assignment to other rooms as well. Other validation checks are performed dynamically during the simulation. If there are any conflicts in dependent relationships, the simulator will indicate this. For instance, if some roles are performing a certain task in a certain room, and some of the roles do not have the permission to enter the room, the simulator will indicate that conflict.

Using this concept AMBOSS is able to handle topological aspects and relationships that may be important in safety critical systems. We see this approach as a first step in this direction, which has to be developed further. We could, however, successfully model the safety-critical process of the take-off process of airplanes from an aircraft carrier, where the location of material and human actors is of vital importance.

5 Analyzing Communication

Deficiencies in the communication between the actors of a socio-technical system are recognized as a major cause for critical incidents and accidents [4]. Therefore the integration of this aspect into the task model represents an important expansion for our approach. A lot of problems in safety-critical socio-technical systems have their roots in communication problems. An improper sequence of task performance, for instance, could be caused by missing communication (too early, too late); a task could be mal-performed by lack of feedback (misunderstanding vital information), which could also be the case when problematic situations are unobservable (wrong display). On a higher level of abstraction, tasks could put an unbalanced strain on actors, because they do not communicate enough.

Some approaches like CTTE [12] or TaskArchitect [17] integrate communication to some degree as a part of the model. However, they do not take into account relevant parameters of the communication, such as triggering mechanisms, criticality, necessity of feedback, kind of medium or content. Within AMBOSS we included the possibility to specify communication and its parameters and developed an appropriate method of analysis.

As a fundamental theoretical model, we choose the information-communication model of Shannon and Weaver. In this model communication is seen as exchange of information (signals) between transmitter and receiver. This excludes a priori a large part of aspects otherwise coming under the term of communication, such as psychological, social, or ethical issues. Basing our concept on information theory restricts communication in our model to the flow of information. In this context, every task and its associated role in a model can be sender and recipient of information. The most important circumstances for good communication (especially important when taking into account safety critical systems) are, that the information is correct, that the information is complete and its transmission cannot be changed by external factors, and that the information is correctly interpreted by the recipient.

The examination of success or failure of communication necessitates considering the following aspects of information flow: the actors (transmitter, receiver), the information itself, the transmission medium, supervising control structures, timing, and the environmental situation. Most of these parameters can be captured in an AMBOSS model, with the exception of the last, which is a very open property. Communication in AMBOSS is dealt with during the modeling phase, and taken into account by the simulator, as mentioned before. Apart from that, however, we developed an analysis process of the communication in socio-technical systems which will be integrated into our approach of the extended task modeling. This process goes beyond the evaluation of single communication paths but provides an overall assessment of the communication situation in the given task model and hints directly at weak points.

The analysis consists of four steps, which we sketch out only roughly here, details can be found in [11]. In the first step, all communication sequences are classified into one of three groups of criticality. This is done based on the critical status of the sender and the receiver which provides a basis for the priority of the fact-finding. Second, we look closer on particular message sequence. This exploration contains a binary evaluation of each sequence from the point of view of both partners. This part of the analysis already shows the most important weaknesses of the communication flow. After identifying the communication weaknesses in the system we start with the qualitative judgment. Based on this judgment we can give a more informative statement about each parameter within the different sequences of the communication flow. Third, the aggregation of the identified weak spots in the system takes place. The analyst is able to recognize which parameters are affected and therefore should be modified to improve the safety of the whole communication flow. Finally, the analyst can specify at which position of the communication flow the system shows deviations from the expected procedure. Finally, it is possible to derive suggestions that are classified along the single communication parameters for a proposed improvement.

By applying this analysis method to real-world case studies, we found out that it is especially valuable for detecting latent errors in a system. These are errors which have been "done" much earlier and in general by completely different people than the ones that suffer from the consequences when actually performing a task. An example is the erroneous adjustment of a perfusor which happened due to a blurred communication protocol between the responsible actors. The definition of the protocol was identified as a latent error during the communication analysis.

Currently the method is based on AMBOSS but is not yet formally integrated. The emphasis up to now was to define the single steps. We will work on a tool, however, to help the analyst to perform the analysis on AMBOSS models.

6 Defining and Finding Patterns

The enhancements of task modeling approaches within AMBOSS presented so far, add more information and detail to a potentially already complex model. The analyst needs to master the complexity of this model and while entering more detail information, he might lose the overview. In situations, where the detail information specified in different places in its combination creates a critical situation the following approach referred to as AmbossA helps to detect these constellations.

AmbossA is a visual query language which enables the user to define certain relationship structures between AMBOSS elements and to search for them within a task model. In AMBOSS all elements have or do not have a connection with one another. For example, a message has a sender and a receiver, a role executes or does not execute a task etc. Hence, a task model can be seen as a complex structure of connections between the single elements. AmbossA allows the user to visually create patterns connecting AMBOSS elements such as tasks, roles, object, messages etc. and thus characterizing relationship structures. Given such a pattern, the system localizes the pattern in the task model. If the pattern represents a pathological or dangerous constellation, or some other type of "weak spot", the analyst is directly guided to these problem areas.

To specify a pattern, AmbossA provides certain visual elements (shown in Fig. 3 on the left), corresponding to the elements which can be used within AMBOSS to specify a task model. These can be linked visually with relations of objects, denoting certain semantic relations, such as a role executing a task, or a message triggering a task (see Fig. 3 on the top right).

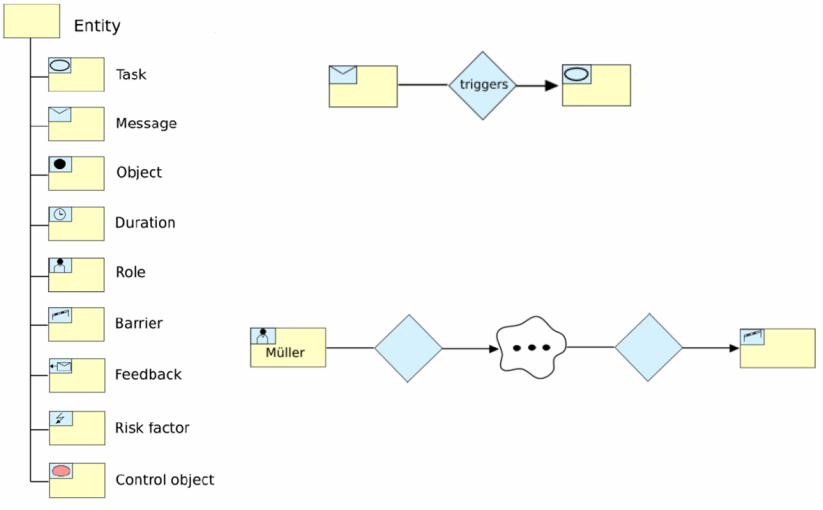

Fig. 3. Visual elements (left) and example relations (right) in AmbossA

Using these elements, the user can find all tasks executed by a certain role, or all messages triggering a certain task etc. In addition to abstract representatives for concrete elements in the task model, the cloud element provides a powerful mechanism for finding certain constellations. If two elements are connected with the cloud, then they are connected "somehow", and all possible paths between the two elements are considered. Hence, the cloud plays the role of a wild card symbol in the diagram language. For example, it is possible to search for any relationship between a specific role ("Müller") and any barrier, as depicted in Fig. 3 on the bottom right.

In combination with the logical operators AND, OR and NOT, the user can express complex constellations of AMBOSS elements and search for them. In the diagram shown in Fig. 4, all situations are found where a message is sent from either two tasks or a role to a target task.

The result of such a query is a table of elements matching the given diagram. In its current version the user can skip through this list and is directly shown the position of the situation in the task model. In addition, the match currently selected in this list is shown as a highlighted substructure in the AMBOSS editor.

The speed of the parser process directly depends on the abstraction level of the relationship pattern. The more abstract the pattern is, the more time the search algorithm needs. The most challenging situation for the algorithm arises if the relationship pattern contains several „clouds". However, for practical purposes we found the runtime behavior perfectly acceptable. More work will have to be done to develop this idea further towards a tool to administer a library of "good" or "bad" patterns, which could then automatically be applied to task models of safety-critical socio-technical systems.

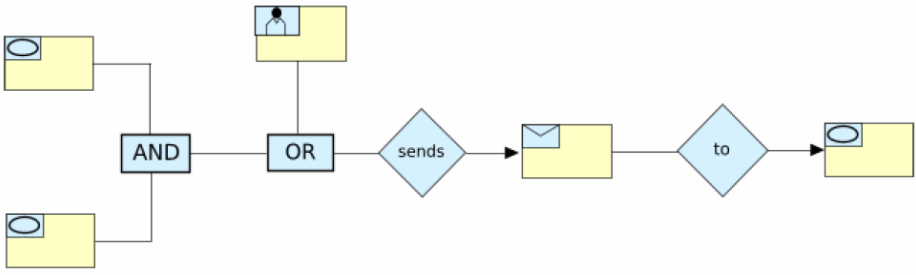

Fig. 4. A visual query in AmbossA

7 Conclusions

The current state of AMBOSS is that the editing component and the simulator comprise all elements mentioned in chapters 3 and 4, and the communication parameters mentioned in chapter 5. The pattern matching algorithm, resp. the visual language AmbossA is fully functional. As mentioned in chapter 5, the communication analysis algorithm is a recent development; the integration with AMBOSS is under way.

Overall, the approach provides a rich set of enhancements over classical task models which are especially useful for safety-critical socio-technical systems. As single concepts they exist in the rich literature on safety-critical systems but they have not been integrated with the concepts of task modeling before. The inclusion of a powerful simulator taking all these elements into account enables the analyst to get an overview of effects such as information flow, barrier dynamics, role distribution, and spatial relations during task execution.

AMBOSS invites enhancements, based on its underlying model structures, as we implemented a programming interface to link new analysis tools to the environment. We exploit this in a first co-operation with the developers of PetShop ([2], [13], and [15]), to link task model simulation and user interface prototyping during runtime.

In our view, the combination of task modeling and safety-critical analysis concepts, as instantiated in AMBOSS, is a promising approach. It takes procedural knowledge of a work process into account and links it to the otherwise unrelated safety aspects. The task model contains knowledge about timing, dependencies of tasks, modified objects, sent messages, etc. – hence it can provide valuable information where targeted safety improvements for those concerned can be implemented.

References

1. Amboss: Homepage of the AMBOSS project, Universität Paderborn, Institut für Informatik (accessed April 2008), `http://wwwcs.upb.de/cs/ag-szwillus/lehre/ws05_06/PG/PGAMBOSS/index.php`
2. Barboni, E., Navarre, D., Palanque, P., et al.: A Model-Based Tool for the Formal Modelling and Simulation of Interactive Safety Critical Embedded Systems. In: Proceedings of HCI aero conference (Demonstration) (HCI Aero 2006), Seattle, USA (September 2006)

3. Baron, M., Lucquiaud, V., Autard, D., et al.: K-MADe: un environement pour le noyau du modèle de description de l'activité. In: Proceedings of the 18th French-speaking conference on Human-computer interaction, Montreal, Canada, April 18-21 (2006)
4. Bellamy, L.J., Geyer, T.A.W.: Development of a working model of how human factors, safety management systems and wider organisational issues fit together. Health and Safety Executive Report, RR543 (2007), http://www.hse.gov.uk/research/rrpdf/rr543.pdf (accessed, June 2007)
5. Biere, M., Bomsdorf, B., Szwillus, G.: Specification and simulation of task models with VTMB. In: CHI 1999 Extended Abstracts on Human Factors in Computing Systems, Pittsburgh, Pennsylvania, May 15 - 20, 1999. ACM Press, New York (1999)
6. Guitare: GUITARE Homepage (Last access: April 2008), http://giove.cnuce.cnr.it/Guitare/
7. Hollnagel, E.: Barrier analysis and accident prevention. Aldershot, UK, Ashgate (2004)
8. Isolde, Isolde Homepage (Last Access: April 2008), http://www.ict.csiro.au/staff/Cecile.Paris/from-cmis/projects/Isolde/scientificProgress.htm
9. Lu, S., Paris, C., Vander Linden, K.: Tamot: Towards a Flexible Task Modeling Tool. In: Proceedings of Human Factors, Melbourne, Australia, November 25-27 (2002)
10. Mefisto: Mefisto Homepage (Last Access: April 2008), http://giove.cnuce.cnr.it/mefisto.html
11. Mistrzyk, T.: Analysis of Communication in Hierarchical Task Models focused on Safety Critical Systems. Dissertation, University of Paderborn, Institute of Computer Science (in print) (2008)
12. Mori, G., Paternò, F., Santoro, C.: CTTE: support for developing and analyzing task models for interactive system design. IEEE Trans. Softw. Eng. 28(8), 797–813 (2002)
13. Navarre, D., Palanque, P., Barboni, E., et al.: On the Benefit of Synergistic Model-Based Approach for Safety Critical Interactive System Testing. In: Winckler, M., Johnson, H., Palanque, P. (eds.) TAMODIA 2007. LNCS, vol. 4849, pp. 140–154. Springer, Heidelberg (2007)
14. Puerta, A.R.: The Mecano Project: Enabling User-Task Automation During Interface Development. In: AAAI 1996: Spring Symposium on Acquisition. Learning and Demonstration: Automating Tasks For Users, pp. 117–121 (1996)
15. Petshop: Petshop Homepage (Last access: April 2008), http://liihs.irit.fr/petshop/; Shannon, C., Weaver, M.: The Mathematical Theory of Communication, University of Illinois (1949)
16. Storey, N.: Safety-Critical Computer Systems. Addison-Wesley, London (1996)
17. Stuart, J., Penn, R.: TaskArchitect: taking the work out of task analysis. In: Proceedings of the 3rd Annual Conference on Task Models and Diagrams, TAMODIA 2004, Prague, Czech Republic, November 15 - 16, 2004, vol. 86, pp. 145–154. ACM Press, New York (2004)
18. Szekely, P., Luo, P., Neches, R.: Facilitating the Exploration of Interface Design Alternatives: The HUMANOID Model of Interface Design. In: CHI 1992 Conference Proc. (1992)